

# LLM Integration Decision

## 1. Overview

The AI Operator project is designed to automate browser-based tasks like form-filling, grocery shopping, or ticket booking while preserving user privacy by running entirely on the local device. A key requirement is an intelligent system that can interpret natural language commands, analyze webpages, and generate structured sequences of browser actions.

After evaluating various language model platforms, we have chosen to integrate the LLM Claude (by Anthropic) alongside Microsoft Semantic Kernel for orchestration and control. This combination provides the natural language planning strength of Claude and the controlled execution environment of Semantic Kernel.

## 2. Chosen Stack: Azure OpenAI + Microsoft Semantic Kernel

### ◆ Claude by Anthropic

Claude 3.5 is a high-performance language model designed for safe and transparent AI interactions. It is particularly strong in multi-step planning, structured reasoning, and instruction following. It supports advanced function-calling formats, making it suitable for translating user commands into machine-readable steps.

### ◆ Microsoft Semantic Kernel

Semantic Kernel is an open-source orchestration engine built by Microsoft to combine LLMs with user-defined functions and memory. It allows us to:

- Break user goals into actionable steps (plan & execute)
- Safely limit the actions LLM can perform (click, type, navigate)
- Integrate external tools like scrapers and browser controllers
- Maintain context/memory across interactions

## 3. Benefits Over Other LLMs & Tools


Compared to alternatives like Azure OpenAI and local models, the Claude + Semantic Kernel stack offers:

- Superior multi-step planning: Claude performs well at translating vague user requests into structured action plans.
- Model safety and transparency: Claude is designed with alignment and instruction safety in mind.
- Function-calling support: Claude supports formats for tool use and JSON-based planning.
- Flexible tool orchestration: Semantic Kernel provides built-in planning support and secure function execution, making it a strong match for Claude.

## 4. AI Operator Workflow in Action(Observe-Plan-Act)

 User Input:

“Add two keto snacks to my Tesco cart.”

 Claude (via prompt or function call) generates:

```
{
  "steps": [
    {"action": "navigate", "target": "tesco.com"},
    {"action": "search", "query": "keto snacks"},
    {"action": "add_to_cart", "items": ["Item A", "Item B"]}
  ]
}
```

 Semantic Kernel then:

- Executes each step safely via browser automation tools (like Selenium)
  - Handles errors and retries
  - Maintains action memory/context
- Claude and Semantic Kernel work together by dividing responsibilities: Claude handles the interpretation and planning of user commands, while Semantic Kernel manages structured, safe execution. When a user provides input, Semantic Kernel sends it to Claude, which returns a step-by-step action plan (e.g., in JSON). Semantic Kernel then executes each step using predefined tools, such as browser automation scripts. This loop continues—updating context and requesting next actions—ensuring accurate understanding from Claude and reliable, rule-based execution through Semantic Kernel.

## 5. Other Options Considered

### A. Azure OpenAI (GPT-4) + Semantic Kernel

- Lacks: Flexible deployment, efficient structured outputs.
- Why not: GPT-4 often produces verbose or less predictable action plans. Claude gives more consistent, step-by-step results for task execution.

### B. OpenAI API (Direct)

- Lacks: Native Semantic Kernel support, strong safety controls.
- Why not: Claude is better aligned for instruction-following and function calling. OpenAI API is more generic and harder to control securely.

### C. Local LLMs (LLaMA, Mistral)

- Lacks: Function-calling support, high-quality reasoning.
- Why not: These models require complex setup and don't match Claude's planning accuracy. They're unsuitable for real-time browser automation tasks.